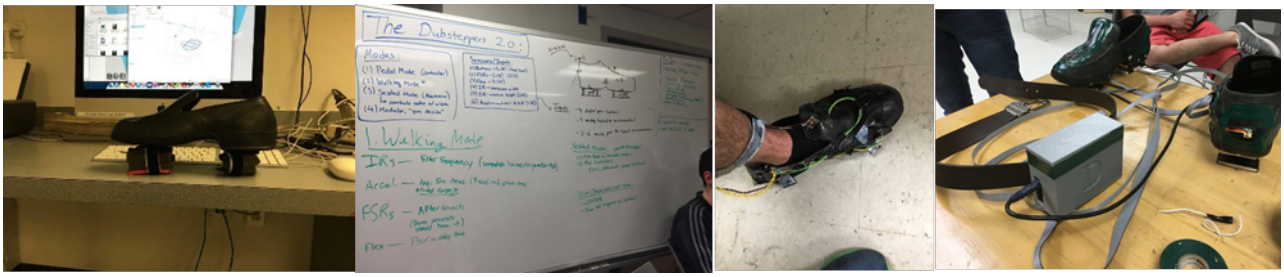
The Dubstep's 2.0
Joey Cirone
5/12/2016

Teammates: Thomas Coons, Jeremy Marcus & Eamon Wick



Concept:

Utilize a pair shoes and a combinations of natural and unnatural gestures to control musical expressions.

The Dubstep's 2.0, the second version of the original Dubstep's, is essentially a MIDI sequence controller with additionally functionality to play selected notes, without a pre-defined sequence. There are four distinct motions that contribute to how the Dubstep's "step" through a pre-programmed sequence of notes. Motions and gestures including walking, bending your ankle, shaking your foot, tapping your toe, tapping your heal, and a fast stomp all translate to musical notes and expression.

How it works:

The Dubstep's 2.0 has 4 distinct modes of playing, making it much more versatile than the original Dubstep's. There are a total of 15 sensors including buttons (x4), force sensing resistors (x4), IR's (x3), flex (x2), and accelerometers (x2). All of this data can be processed to map to MIDI data, thus creating very unique and versatile musical expression.

Mode 1, "walking mode", is very similar to the original Dubstep's. This mode utilizes the four distinct motions that contribute to somebody walking. Pressing down on the heel triggers note-on events, and lifting up off the toe triggers note-off events, simulating a walking motion. Triggering a note-off is not possible once the heel triggers a note-on until the toe sensor is activated, allowing for sustain. Re-attacking a note with the heel once the toe has been triggered is possible. Pressing down hard on a toe or heel, will trigger MIDI after touch, which can be mapped to many MIDI expressions such as volume control, LFO rate, or filter frequency. The accelerometer gets velocity data which is mapped to how fast you make a step, the more velocity, the louder the note is played. Lifting each shoe, measured by the IR sensor on each heel, changes the LFO rate of each note. Lastly, the Flex Sensor changes the filter frequency of the notes. The more you bend your ankle, the greater effect the filter frequency will have. Various songs and notes can be used to "step" through the sequence. In our demo, "In The Hall of The Mountain King" was played. Changing the sequence is as easy as loading in a text file of MIDI note numbers in the correct order.

Mode 2, "Seated Theremin Mode", allows the player to select notes based on the distance between each shoe. The IR sensor on the inside of the foot measures the distance between each shoe and maps that distance to a note in a major scale. Because of the logarithmic relationship between distance and voltage of the IR's, the distance between the 7^{th} to 8^{th} note is much greater than the 2^{nd} and 3^{rd}. To trigger a note, the player pressed either the right toe or heel button. The left shoe should be stationary

to act as a reference. After touch can be triggered by pressing down on the right foot, which compresses the FSR, which can be mapped to many MIDI expressions such as filter frequency and LFO rate.

Mode 3, "Drum Loop Mode", maps four drum tones to each of the four buttons on the shoe. This mode attempts to create a drum machine using the Dubstep's. We were not able to effectively complete the looping aspect of this mode, but the main sounds are present. If the looping was available, the player would be able to create a drum beat, and then switch to Mode 1 or Mode 2.

Mode 4, "Modular Mode", is the most unique part of the Dubstep's 2.0. The user has the ability to map sensors to MIDI commands, thus creating a true customizable MIDI controller. Buttons, IR's, accelerometers, flex, and FSR's can all be mapped to MIDI commands such as note-on, note-off, after touch, velocity, pitch, and countless others.

Design Goals:

The Dubstep's 2.0 goal was designed to be a very modular instrument that can be customizable by the user with 3 default modes. The default modes allow the instrument to be played with easy, more natural, settings and the Customizable Mode should be used for the advanced and daring players for maximum creativity. Ultimately, the Dubstep's 2.0 can be a portable guitar foot petals. The player could walk around the stage and trigger various "guitar pedal sounds" by mapping sensors and gestures to a MIDI commands. The Dubstep's design was to be very compact, portable, customizable and easy to use. The versatility of this instrument is infinite and is only limited by a player's creativity. Unfortunately, portability was not completely functional because our wireless technology was too difficult to implement in the time given.

Parts Used: An Ode to Sensors

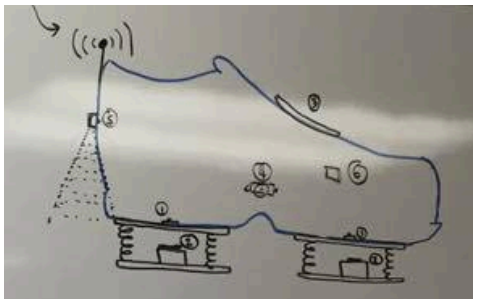
- (x4) Force Sensing Resistors (FSR's) [sparkfun SEN-09376] used to detect pressure on the heel and toe for each shoe. Placed on the outside spring construction of each toe and heel.
- (x2) 4.5" Flex sensors [sparkfun SEN-08606] used to detect an ankle role or flex. Placed on the flap of each shoe.
- (X4) Assorted Buttons [sparkfun SEN-10302] used to detect toe and heel press. Attached to inside soles of both shoes.
- (X3) Short Ranged Infrared Sensors [sparkfun Sharp GP2Y0A41SK0F] to detect distance between shoes and ground of each shoe. Placed on the inside of one shoe and the other two placed on the back heel of the shoe facing downwards.
- (x2) Accelerometers [sparkfun SEN-9269] to detect velocity of each foot step. Placed on the outside front of each shoe.
- (x2) XBee 1mW for wireless transmission [sparkfun SEN-11215]. One connected to the Computer via an Explorer and another to the Arduino via a Shield.
- (x1) XBee Explorer Dongle [sparkfun SEN-11697].
- (x1) XBee Shield [sparkfun SEN-12847].
- (x1) Arduino Mega 2560

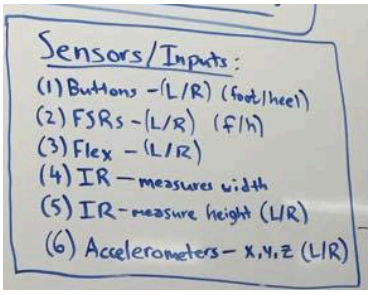
The Dubstep's 2.0
Joey Cirone
5/12/2016

Teammates: Thomas Coons,
Jeremy Marcus &
Eamon Wick

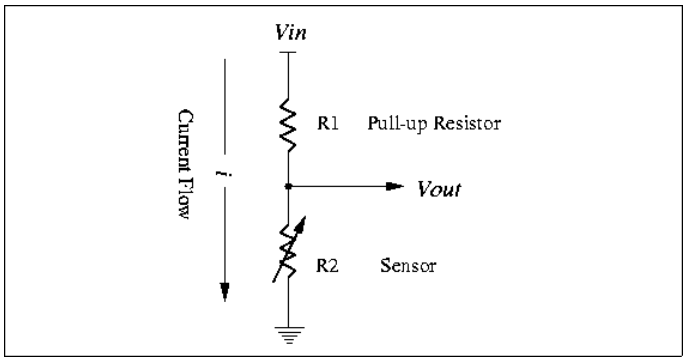
Construction:

The construction of the Dubstep's was much more difficult than anticipated. Because there were 7 or 8 sensors on each shoe, there was a total of 22 inputs to the Arduino. To keep construction neat and clean, we used ribbon wire. The other connection of the ribbon wire went to a single circuit board with any necessary resistors and regular gauge wire to connect to the Arduino. Below is the sensor construction schematic for each shoe.





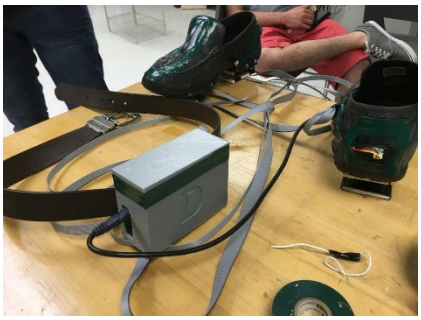
Most sensors except for the flex and FSR just needed inputs. The FSR and bend sensor are variable resistors. To read off a value, a voltage divider was used. Below was the circuit used for these sensors.

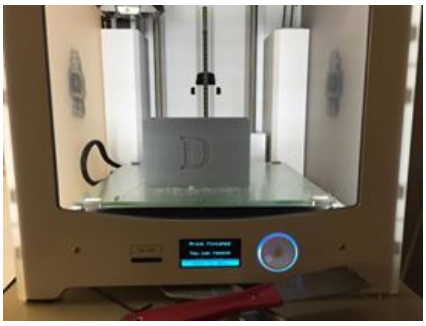


Vin was 5V from the Arduino and R2 was the FSR or Bend Sensor. A 10K resistor was recommended for the R1 value for each sensor. The values for the FSR's into the Arduino ranged from 1 to 1023 and the bend sensor from about 470 to 850 which gave us a lot of room to manipulate data. Each sensor used 5V from the Arduino except for the Accelerometer which used 3.3V.

To create an after touch effect and feedback from the FSR's, we created a spring contraption as seen above. Two pieces of plywood with 4 springs created one of these contraptions. The FSR was placed inside of this with a foam insulation.

To hold the Arduino and circuit board in one compact place, we created a 3D printed belt box that could be secured by threading a belt through it. Below are two pictures of the construction.





Max & Reason:

The Max patch was the most complicated part of this project. An initial patch was made that created a state machine of the various gestures the Dubstep's could make. States included, heel up, heel down, toe up, toe down, heel to toe, toe to heel and smooth transition.

Calibration was the key for figuring out the state of the sequence. An averaged reading was used to prevent jitter. The brains of the controller was the state machine patch and then each other patch was basically a module of various sequences. The reason rack changed based on what sequence was used.

The modularity and friendly user interface of the Max patch was the key design factor and choice for the Dubstep's. For "Seated Theremin Mode", a virtual Piano Keyboard displayed what notes were being played which also gave the ability for the player to select the key of the Major Scale to be used. Also, the main screen of the Max patch enabled the user to map sensors to MIDI controls just with a click and drag motion by connecting these objects.

Reason was fairly simple. We used many modules such as the Combinator, Subtractor, and other modules. After finding the write MIDI control numbers, it was easy to map sensor value from MAX to change the Reason module controls.

The Future of the Dubstep's 2.0:

First and foremost, if more time was given, we would have made the Dubstep's 2.0 wireless using the XBee technology. Unfortunately, this was much more difficult than anticipated. The Arduino2Max patch was not able to send a message to the XBee via serial port which was essential for synching up data from the Arduino. In an attempt to fix this, we transmitted data from the Arduino every 15 milliseconds, as the Max protocol required. Unfortunately, there were many sync issues and the Arduino port data did not process correctly. Although wireless did not happen, the neat construction of the Dubstep's 2.0 as well as the 3D printed box and 16 foot USB cable allowed the player to still walk around.

Because there are so many sensors on each shoe, a plethora of data is being sent and processed by the Arduino. With different processing, this data can be interpreted and implemented in many different ways. A very exact pedometer can be implemented with both shoes to not only detect step count, but also how much pressure the user puts on the toe and heel which can be used for medical use. Ultimately, these shows could be worn by dancers to trigger music based on the gestures they use. For this reason, the Dubstep's 2.0 are not only a MIDI controller, but an infinite possibility of implementations only limited to your creativity.